《AGI 逻辑导论》

YKY

January 9, 2021

Summary

- 描述一种可以完整地解决 AGI 的逻辑

Contents

1 Background

我们想 **训练** 一个智能系统，训练 是一个 **机器学习** 的过程，也是一个 **optimization** problem, 目标是将 **长期的奖励总和** 最大化：

$$\text{maximize: } \int_0^\infty R\,dt \tag{1}$$

where $R(t) =$ reward at time t. $\int_0^\infty$ 表示 计算 累积奖励的 **time horizon**. （我使用了微分的形式，实际应用通常是离散形式，但两者基本一样，不必深究）

俗语说「棋屎贪吃卒」，在开局初期吃卒，可能导致 N 步之后被将死，这是**愚蠢**的行为。所以 (1) 式 令 系统必需顾及长远的利益，遂迫使它学习 **智慧**。

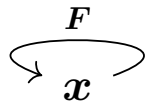
Architecturally, the AI is a **dynamical system** that constantly updates its "state" $\boldsymbol{x}$ via: *

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) \tag{2}$$

或者用离散形式表示：

$$\boldsymbol{x}_{t+1} = \boldsymbol{F}(\boldsymbol{x}_t) \tag{3}$$

$\boldsymbol{F}$ 叫作 transition function. 或者更形象地表示：



(4)

Our goal is to **learn** the function $\boldsymbol{F}$, implemented as a **deep neural network**. $\boldsymbol{F}$ 包含智能系统内的所有**知识**。

* Part of the state $\boldsymbol{x}$ contains **sensory input** and **action output** that allow the AI to interact with the external environment.

1.1 飞机比喻

最早的飞机，使用 **平面** 做机翼，而不是模仿**飞鸟**的拍翼。它使用 **螺旋桨** 作为动力，因为当时最强的动力装置 是**内燃引擎**：

$$\neq \quad \text{  } \qquad (5)$$

深度学习是现时**最强**的学习算法，它可以学习非常复杂的 非线性函数。问题是怎样利用这件「武器」。我提出的 architecture 就是 (4)，这也是 Richard Sutton 提出的基於 强化学习 的模型。*

时至今日，飞机仍然叫 “plane”，而这个设计基本上奠定了 100 多年以来 飞机的模式。在技术的进化上，这种主导模式的现象称为 dominant design.

我提议 利用 **逻辑结构**，**约束 $\boldsymbol{F}$ 的搜寻空间**（这叫 inductive bias），令它可以更快地学习到人类水平的智能。这是不是达到 AGI 的最好的方法呢？不实践是无法知晓的。Sutton 甚至认为，不需要人为的 bias，纯粹增加**计算力**就可以了。

Richard Sutton (1949-)

* AGI 的架构还可以包括 episodic memory 等部分，现在我们考虑的是 minimalist architecture. 如果不用高度抽象的理论，很多时会迷失在支节里。

2 Structure of logic

The central tenet of my theory is that the state $\boldsymbol{x}$ of the AI system is consisted of **logic propositions** and that $\boldsymbol{F}$ plays the role of the **logic consequence** operator $\vdash$:

$$\boxed{\text{propositions}} \overset{\boldsymbol{F}}{\longmapsto} \boxed{\text{propositions}} \tag{6}$$

So our goal now is to elucidate the structure of $\vdash$. Currently the most elegant formulation is given by **categorical logic** or **topos theory**.

我发觉 我是一个擅长於 "synthesize" 的人，意思是我会看很多书，然后将各种分散的 ideas 融合成一个 内部协调 的理论（当中大部分 ideas 不是我原创的）。

在接下来的篇幅，我会勾划一个 对於 AGI 来说是完整的 逻辑理论，而这理论 的中心思想 就是 Curry-Howard isomorphism....

3 Curry-Howard correspondence

Curry-Howard isomorphism 是一个很深刻的思想，如果不小心的话 甚至会觉得它讲了等於没讲。

简单来说：当我们做 逻辑思考时，表面上有一种语法上 (syntax) 的形式，即 $A \Rightarrow B$:

$$\begin{array}{cc} \boxed{\text{logic}} & A \Longrightarrow B \\ \hline \boxed{\text{program}} & \blacksquare \overset{f}{\longmapsto} \blacksquare \end{array} \tag{7}$$

而在这 语法「底下」，还有一个 **运算**，它可以看成是执行 **证明** (proof) 的工作，它将 A 的证明 map 到 B 的证明（为了避免符号累赘，我将这些 "proof witness" 都记作 $\blacksquare$，但它们每个是不同的）.

一个传统的数学函数，例如 $f(x) = x + 2$ 用我们惯常的符号表示为：

$$\begin{array}{c} f : \mathbb{R} \longrightarrow \mathbb{R} \\ \hline x \longmapsto x + 2 \end{array} \tag{8}$$

这不是新的。类似地，一个逻辑式子：

$$x \text{ 是偶数} \implies x+2 \text{ 是偶数} \tag{9}$$

也不是新的。但如果将「x 是偶数」这个**命题**，看成是一个**类型**或**集合**，里面有个**证明** (witness)，这个看法是新的：

$$\frac{\boxed{x \text{ 是偶数}}}{\blacksquare} \tag{10}$$

This is called the **Brouwer-Heyting-Kolmogorov (BHT) interpretation**. 由於这个想法比较 subtle，它不断被重复发现很多次，命名者可以包括：Brouwer-Heyting-Kolmogorov-Schönfinkel-Curry-Meredith-Kleene-Feys-Gödel-Läuchli-Kreisel-Tait-Lawvere-Howard-de Bruijn-Scott-Martin-Löf-Girard-Reynolds-Stenlund-Constable-Coquand-Huet-Lambek ....

根据 HoTT (homotopy type theory)，一个命题 可以**有**或**没有**证明；如果有，则它的证明都是一样的，所以 经典逻辑命题 只可以取值 **真**或**假**。我的理论推断：模糊逻辑的取值 $\in [0,1]$ 是因为 fuzzy 命题的「证明」可以有很多个，而 fuzzy 命题的 **真值**是由 支持 / 反对 的证明的**比例**决定的（见 §7）。

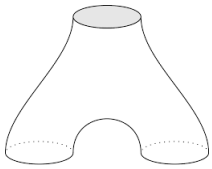
John Baez (1961-)

从另一角度看，Curry-Howard isomorphism 可以看成是 某些 **状态**（states，例如 A）和状态之间的 **转换** （transitions，例如 $\overset{f}{\mapsto}$）之间的 **对偶**。而这种 对偶 不断在 截然不同的范畴里出现：

logic	**computation**	**category theory**	**physics**	**topology**
proposition	type	object	system	manifold
proof	term	morphism	process	cobordism

(11)

前两个就是 Curry-Howard，第三个 是 Lambek 加上去的，其馀的来自 John Baez & M. Stay 的论文：*Physics, Topology, Logic and Computation: a Rosetta stone* [2010]. 例如在 physics 里面是 Hilbert space 和 operators 的对偶；在 topology 里面，cobordism 的著名例子就是这个 “pair of pants”：



(12)

In string theory，它表示上面的 strings 变成下面的 string 的「时间过程」。

3.1 Type theory

描述 **program** 或 **computation** 的语言叫 type theory. 例如在一般的 编程语言 里可以有这样的一句：

```
define length(s: String): Integer = { .... }
```

(13)

意思是说 length() 是一个函数，输入 String，输出 Integer.

在数学里 我们描述 **函数** 时会用：

$$f : A \to B \tag{14}$$

这个表达式其实就是 type theory 的一般形式：

$$\overbrace{t}^{\text{term}} : \overbrace{T}^{\text{type}} \tag{15}$$

而这个 notation $t : T$ 其实也可以写成 $t \in T$（但不正统而已）。

换句话说，types 就是 **集合**，terms 是集合中的 **元素**。

更一般地，一个 type theory 的句子 可以包含 type **context**：

$$\overbrace{x : A}^{\text{context}} \vdash \overbrace{f(x) : B}^{\text{type assignment}} \tag{16}$$

意思就像在 program 的开头 “declare” 一些 变量 的类型，然后 program 就可以被 **赋予** 后面的 类型。

这个 $\vdash$ 的过程 称为 **type assignment**，而这就是 type theory 做的全部工作。

λ-calculus

在一个 program 里，除了定义 类型，还需要定义 **函数**。这件工作是由 λ-calculus 负责。

λ-calculus 可以定义函数 而不需要提及它的「名字」。例如，用数学式表达：

$$f(x) \triangleq x^2 \tag{17}$$

它的 λ-表达式就是：

$$f \triangleq \lambda x.\ x^2 \tag{18}$$

注意：在 λ-表达式里，不需要提到 f 的「名字」。

λ-calculus 是由 Alonso Church 发明，目的是研究数学上 **substitution** 的性质。Substitute 是每个中学生都懂得做的事，但要用数学表达出来却是出奇地麻烦。

同时，Church 发现 λ-calculus 是一种「万有」的计算形式，和 **Turing machines** 等效。「AI 之父」John McCarthy 用 λ-calculus 发展出 **Lisp** 语言，它是所有 functional programming language 的鼻祖。

Curry-Howard correspondence

在 Curry-Howard 对应下，type A 就是 逻辑命题 A，type A 或 集合 A 里面的元素 是其 **证明** (proof, or proof witness)。

而，$A \Rightarrow B$ 也是 逻辑命题，它对应於 the function type $A \to B$，也可以写作 B^A，而这个 type 或 集合 里面的 元素 就是一些 函数 $f : A \to B$. 如果 有一个这样的函数存在，则 type $A \to B$ 有「住客」(inhabited)，换句话说 $A \Rightarrow B$ 有 **证明**。

3.2 Intuitionistic logic

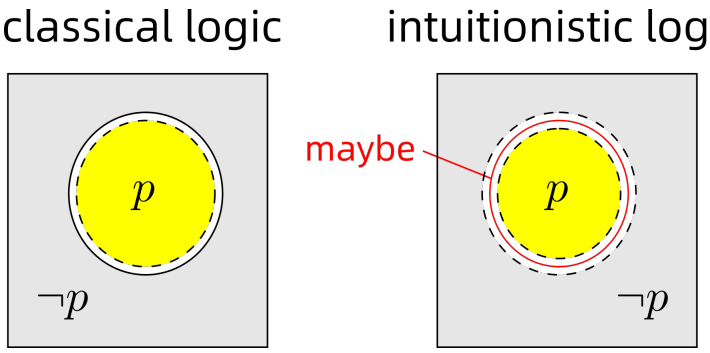
Curry-Howard isomorphism 揭示了 type theory 和 **intuitionistic logic** （直觉主义逻辑）之间的关系。这种逻辑的特点是没有 **排中律** (law of excluded middle, LEM)，或者等价地，**double negation**，即 $\neg\neg p \Rightarrow p$.

排中律 是说：$p \vee \neg p$ 是 恒真命题。但在 直觉主义 逻辑中，$p \vee \neg p$ 表示 p 的证明 **或** $\neg p$ 的证明，但有时候这两者都不知道（例如 现时仍未找到证明，或者不可能找到证明）。

附带一提：人们惊讶地发现，在直觉主义逻辑下，axiom of choice $\Rightarrow$ law of excluded middle. 换句话说，axiom of choice 和 直觉主义 也有内在的矛盾。

Topological interpretation

在 拓樸学 里，一般用 **open sets** 表示空间中的子集（这习惯起源自 Hausdorff 时期）。但 p 的 **补集** $\bar{p}$ 并不 open，所以要将 $\neg p$ 定义为 p 的补集的 **interior**，即 $\neg p \triangleq \bar{p}^{\circ}$. 於是 $p \cup \neg p \neq$ Universe: *



(19)

3.3 Higher-order logic

Propositional logic 的意思是：只有命题，但忽略任何 **命题内部** 的结构。

假设 p, q 是命题，命题逻辑的基本运算 就是 $p \wedge q, p \vee q, p \Rightarrow q, \neg p$.

First-order logic 的意思是：容许 这样的方法 构成 命题：

$$\overbrace{\text{IsHuman}}^{\text{predicate}}(\overbrace{\text{John}}^{\text{object}}). \tag{20}$$

Predicate 的意思是 **谓词**；谓词 是一些「有洞的命题」，它们被填入 objects 之后就变成完整的命题。类似地可以有 **多元**的 predicates，例如：

$$\text{Loves}(\text{John}, \text{Mary}). \tag{21}$$

* diagram from the book: *Classical and Non-classical Logics – an introduction to the mathematics of propositions* [Eric Schechter 2005], p.126.

First-order 指的是：$\forall, \exists$ 这些 **量词** 可以 **作用** 在 objects 的类别上，例如（Mary 人见人爱）：

$$\forall x.\ \mathrm{Loves}(x, \mathrm{Mary}) \tag{22}$$

但 first-order logic 不容许 量词 作用在 predicates 的类别上，除非用 second-order logic.

一个 二阶逻辑的例子是「拿破仑 具有一个好将军应该具备的所有特质」：

$$\forall p.\ p(\mathrm{Good\ General}) \Rightarrow p(\mathrm{Napoleon}). \tag{23}$$

注意 p 是在 predicates 的类别之上量化的。

3.4 旧式 logic with type theory

Type theory 的历史还可以追溯更早。它起源於 Russell 为了解决 **逻辑悖论**，例如：「一个只帮自己不理发的人理发的理发师帮不帮自己理发?」这些 逻辑悖论 根源是在於：定义一样东西的时候，中途 **指涉** 了这个东西本身。这种不良的定义称作 **impredicative**. 为了避免不良定义，每个东西出现之前必需「宣告」它的类型，这就是 type theory 原来的目的。

在 Curry-Howard isomorphism 未被重视之前，有一种更简单地 用 type theory 定义 逻辑的方法。在这种方法下，逻辑命题 $p, q, p \wedge q$ 等 **直接用** terms 定义，而不是像 Curry-Howard 那样，逻辑命题 = types，证明 = terms.

在这情况下 type theory 处理的是 (first- or higher-order) predicate logic 的方面。这是说，例如：

$$\mathrm{IsHuman}(\mathrm{John}) \tag{24}$$

里面 IsHuman 是一个 函数 term，它输入一个 物体，输出它是不是「人」的真值 (truth value) $\in \Omega = \{\top, \bot\}$. 因此 IsHuman 是一个 类型为 $\mathrm{Obj} \to \Omega$ 的 term.

这种做法没有容纳 Curry-Howard isomorphism 的馀地。如果要做到后者，需要的是 Martin-Löf type theory....

3.5 Martin-Löf type theory

根据 Curry-Howard, 下面的 $A \Rightarrow B$ 是一个 逻辑命题，因而是一个 type:

$$\overbrace{\underbrace{\text{Human (Socrates)}}_{\Omega}}^{A} \Rightarrow \overbrace{\underbrace{\text{Mortal (Socrates)}}_{\Omega}}^{B} \tag{25}$$

但另方面，Human() 和 Mortal() 这两个 predicates 也需要借助 type theory 来构成命题，它们也是 types. 红色 $\to$ 和 蓝色 $\Rightarrow$ 的两个层次 是完全不同的 两码子事，但因为 Curry-Howard 而被逼 挤在一起。这就使得 type theory 好像「一心不能二用」。

在 "simple" type theory 里面可以 构造:

- sum type $A + B$
- product type $A \times B$
- function type $A \to B$

分别对应於 直觉主义逻辑的 $\vee, \wedge, \Rightarrow$. 这些是在 **命题逻辑** 层面的，已经「用尽」了 type theory 的法宝。

但 Human(Socrates) 也是由 Human() 和 Socrates 构成的命题，这构成的方法是用一个 arrow $\to$，但已经没有 arrow 可用。

Martin-Löf 提出的解决方案是 引入新的 **type constructors**:

- **dependent** sum type Σ
- **dependent** product type Π

Dependent sum $\sum_A B$ 里面 B 的类型 depends on A. 整个 family of A 的 $+$ 的结果变成类似 product $A \times B$.

Dependent product $\prod_A B$ 里面 B 的类型 depends on A. 整个 family of A 的 $\times$ 的结果变成类似 exponentiation B^A.

Dependent products can be used to define **predicates** such as Human() and Mortal(). They are of type Obj $\to \Omega = \Omega^{\text{Obj}} = \prod_{\text{Obj}} \Omega$. [*]

一个很漂亮的结果是：如果用 $\sum_A B$ 和 $\prod_A B$ 定义 逻辑命题，则这些 types 如果被 inhabited 的话，分别对应於 $\exists A.B(A)$ 和 $\forall A.B(A)$. 这是因为：如果 $A \times B$ inhabited，表示**至少存在**一个 $B(A)$；而如果 B^A inhabited，则存在一个函数，将**任意的** A send to B.

Per Martin-Löf (1942-) was the first logician to see the full importance of the connection between intuitionistic logic and type theory.

Per Martin-Löf (1942-)

3.6 Arithmetic-logic correspondence

很多人都知道，经典逻辑中 $\wedge, \vee$ 对应於 **算术运算** $\times, +$（也可以看成是 fuzzy logic 的 $\min, \max$.）其实这就是 George Boole 尝试将 逻辑 变成 某种代数 的原因。

较少人知道的是 $A \Rightarrow B$ 也对应於 B^A: [†]

A	B	$A \Rightarrow B$	B^A
0	0	1	$0^0 = 1$
0	1	1	$1^0 = 1$
1	0	0	$0^1 = 0$
1	1	1	$1^1 = 1$

(26)

[*] Note that "objects" here mean logic objects, not objects in category theory.

[†]其中 0^0 是「不确定式」，但按照 组合学 惯例可以定义为 1.

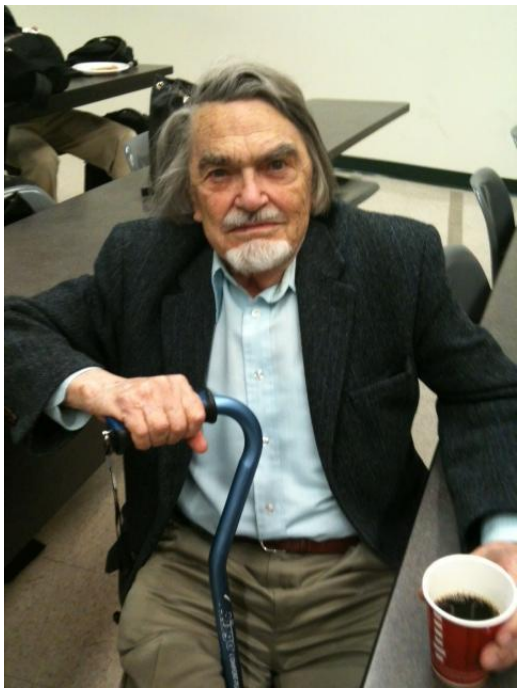
这个惊奇的「巧合」似乎再一次证实 Curry-Howard correspondence 是正确的；特别地，它意味 $\Rightarrow$ 应该看成是 **函数**，即所谓 "functional interpretation of logical deduction."

更详细观察，table (26) 里面 A 和 B 的 truth values 可以看成是它们的 types 有没有 **inhabitants**. Type A 的 inhabitant 就是它的证明 $\blacksquare$，没有证明就是 $\varnothing$. 或者推广到：命题 A 的真值 $= A$ 的 type 作为 集合 的 **cardinality**; The truth valuation of $A = |A|$. 这样看，$A \Rightarrow B$ 的真值 就是 $|B^A|$, 亦即是从 $\{\blacksquare\}$ 或 $\varnothing$ 到 $\{\blacksquare\}$ 或 $\varnothing$ 的 map, 而这个 map 只有在 $\varnothing \mapsto \{\blacksquare\}$ 的时候是空集（不可能）。

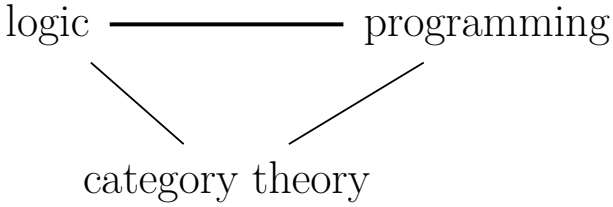
这个观察 可以推广到 fuzzy logic (§7.1) 和 strict implication $A \multimap B$ (§6.5).

4 Topos theory

Joachim Lambek (1922-2014)



将 type theory 对应到 category theory，这工作是 Lambek 做的，於是完成了 Curry-Howard-Lambek 的「三位一体」:



(27)

Topos 的重要意义在於 它是一个可以用来 **进行逻辑运算** 的范畴，关键在於它可以表达 **子集** 的概念，或更一般地叫作 **sub-objects**.

一个 topos $\mathcal{C}$ 里面存在 sub-object classifer Ω 使得 $X \to \Omega \cong$ sub-objects of X. 换句话说 X 的 子集 可以用 $X \to \Omega$ 这个映射来 **represent**.

在 **Set** 这个 topos 里面，Ω 是一个有**两个**元素的集合，可以记作 $\{\top, \bot\}$. 那么 $X \to \Omega$ 就是一些 **命题**，例如 X 是人的集合，则 $X \xrightarrow{\text{mathematician}} \Omega$ 定义哪些人是数学家。

Topos theory 里面最重要的 commutative diagram 是这个：

$$\begin{array}{ccc} X & \xrightarrow{!} & 1 \\ {\scriptstyle m}\downarrow & & \downarrow{\scriptstyle \text{true}} \\ Y & \xrightarrow[\chi_m]{} & \Omega \end{array} \tag{28}$$

其中：

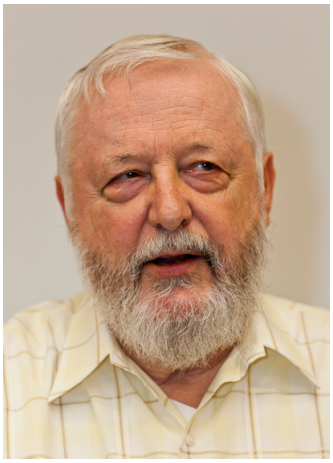
- $X \xrightarrow{!} 1$ 是个 unique arrow，它将 集合 X **整个地**映射到 1. 而 1 是 terminal object，它的定义就是说，通向它的箭咀只能有一个。
- $1 \xrightarrow{\text{true}} \Omega$ 在 $\top$ 和 $\bot$ 之间选择 $\top$, 因此叫 “true” arrow.
- $X \rightarrowtail^{m} Y$ 是 **monic** arrow，特别地，在 **Set** 里面它就是 **inclusion** map，即 $X \hookrightarrow^{m} Y$. 它表示 X 是 Y 的**子集**，$X \subseteq Y$.
- $Y \xrightarrow{\chi_m} \Omega$ 是集合论中熟悉的 **characteristic function**, 当元素 $e \in X \subseteq Y$ 时，$\chi(e)$ 取值 1，否则为 0. 正是 χ_m 的存在令这幅图 commute. χ_m 也记作 $\ulcorner m \urcorner$.

不熟悉基本范畴论的读者，我非常推荐看一看《Conceptual Mathematics》这本书，写得连中学生也可以看懂，而作者之一的 Lawvere 正是 topos 理论的创始人。

从 **Set** 的角度看，这个 diagram 很易理解，但 topos 的好处是它可以将这些逻辑概念 **generalize** 到比 **Set** 更一般的范畴。

Topos 理论的重要性 在於 它用 category 的语言 **重新表述**了 集合论的整个基础。特别地，逻辑学中的符号，例如 $P(x), \forall x, \exists x$, 表面上看似无法用范畴论表示，这正是 Lawvere 惊人的成就。

William Lawvere (1937-)



再看一次 图 (28):

$$\begin{array}{ccc} X & \xrightarrow{!} & 1 \\ {\scriptstyle m}\downarrow & \lrcorner & \downarrow{\scriptstyle \text{true}} \\ Y & \xrightarrow[\ulcorner m \urcorner]{} & \Omega \end{array} \tag{29}$$

右边的 $\begin{smallmatrix} 1 \\ \downarrow \\ \Omega \end{smallmatrix}$ 称为 **generic subobject**. 它的 **pull back** square 用 $\lrcorner$ 记号表示。而左边的 $\begin{smallmatrix} X \\ \downarrow \\ Y \end{smallmatrix}$ 则是一般的 sub-object. We say that the **property** of being a sub-object is **stable under pullbacks**.

4.1 $\wedge$ and $\Rightarrow$ in a topos

Material implication 纯粹 compare truth values, 呢样嘢 fuzzy 之后 唔妥。

Exponentiation B^A 去咗边？

4.2 $\forall$ and $\exists$ as adjunctions

Let Forms($\vec{x}$) denote the set of formulas with only the variables $\vec{x}$ free.

Then there is a trivial operation of adding an additional dummy variable y:

$$* : \text{Forms}(\vec{x}) \to \text{Forms}(\vec{x}, y) \tag{30}$$

taking each formula $\phi(\vec{x})$ to itself.

It turns out that $\exists$ and $\forall$ are adjoints to the map $*$:

$$\exists \dashv * \dashv \forall \tag{31}$$

4.3 Classifying topos $\leftrightharpoons$ internal language

问题是 witness 是什么？例如「匙羹在杯内」是因为其他 视觉 propositions 得到的结论。它是 true 这一点是重要的，但它的 "intension" 更重要。或者说 proof 过程中处理的是 proof objects.

这个 proof object 它除了是一件 syntactic 的东西之外还可以是什么？或者说 map 的 **domain** 是命题，但被 map 映射的是 evidence?

想认识一个**范畴**，最重要的是问：它的 objects 是啥？它的 morphisms 是啥？

Lambek 给出的对应是：

- types $\leftrightsquigarrow$ objects
- terms $\leftrightsquigarrow$ morphisms

We have the following transformations betweeen two formalisms:

$$\boxed{\text{topos}}\ \mathcal{C} \underset{\text{classifying topos}}{\overset{\text{internal language}}{\rightleftarrows}} T\ \boxed{\text{type theory}}\ . \tag{32}$$

In other words,

$$\mathcal{C} = \mathcal{C}\ell(T), \quad T = \mathrm{Th}(\mathcal{C}). \tag{33}$$

4.4 Sheaves and topos

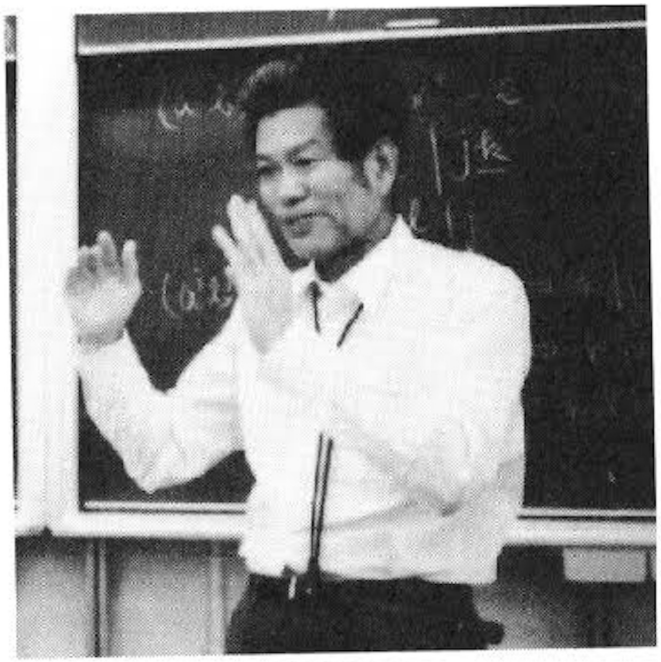
Some **Set**-valued functors are **representable**, ie, isomorphic to a hom-functor.

Functors $\mathcal{C} \to \mathbf{Set}$ are called **pre-sheaves** on $\mathcal{C}$.

Sheaves capture "indexing".

4.5 Yoneda lemma

米田 信夫 (1930-1996)



在一个范畴 $\mathcal{C}$ 里面，考虑 其中一个物体 A 到其他物体的 morphism, $A \to \bullet$. 这可以说是，透过 A「看」其他物体的方法。

例： 在 **Set** 里面，$1 \to X$ 是用 终点物体 1「看」其他物体，看到的是集合的**元素**。

例： 映射 $\mathbb{R} \to X$ 是 空间 X 中的 **曲线**，可以说 $\mathbb{R}$「看到」曲线。

例： 在 ordered set $(\mathbb{R}, \leq)$ 里面 物体 $0 \to x$ 可以「看到」x 是不是 **positive**.

类似地，可以考虑 对偶 的情况，$\bullet \to A$ 是其他物体怎样「看」A 的方法。

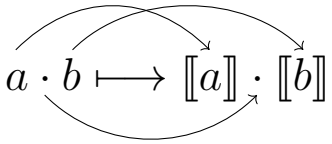
例： 在 **Set** 里面，$X \to 2$ 是其他物体「看」2 的方式，得到的是 X 的**子集**, $\wp(X)$.

例： 在 **Top** 里面，2 包含一个 open set 和一个 closed set，$X \to 2$ 得出的是 X 的**开子集**，$\mathrm{Opens}(X)$.

How sheaves gives rise to representables.

4.6 Model theory, functorial semantics

We interpret formulas in a topos $\mathcal{E}$ by assigning each an **extension**. This is called **internal** semantics.



(34)

4.7 Generalized elements and forcing

一个 逻辑命题 ϕ 可以看成是由 某论域 $A \xrightarrow{\phi} \Omega$ 的函数，其中 $\Omega = \{\top, \bot\}$.

也可以说：命题 $\phi(x)$ 是真的，其中 x 是 A 的**元素**。In category theory, we use the terminal object 1 to "pick out" elements of A, as follows:

$$1 \xrightarrow{x} A \xrightarrow{\phi} \Omega. \tag{35}$$

In **Set**, 任意一个由 1 出发的函数 $x : 1 \to A$ 可以直接看成是 A 的「元素」。

但如果我们用另一个 论域 C 取代 1，换句话说：

$$C \xrightarrow{x} A \xrightarrow{\phi} \Omega. \tag{36}$$

这样的 $x : C \to A$ 叫作 A 的 **generalized element**.

另一个术语是：C **forces** $\phi(x)$, notation $C \Vdash \phi(x)$.
或者说 $\phi(x)$ is true **at stage** C （这术语来自 possible-world semantics）.

4.8 Kripke-Joyal / external semantics

External semantics describe which generalized elements satisfy each formula.

An "internal" way to interpret type theory in a topos is where a formula ϕ in context $x_1 : A_1, ..., x_n : A_n$ is interpreted as a subobject of $A_1 \times ... \times A_n$. This has the disadvantage that the most pleasant illusion of "elements" is totally lost.

用 Kripke-Joyal semantics 可以挽回 generalized elements 的诠释方法。

Generalized element 的意思是 $I \xrightarrow{a} A \xrightarrow{\phi} \Omega$, 记作 $a \Vdash \phi$.

4.9 Cohen's (dis)proof of Continuum Hypothesis

这一节和 AGI 无关，但因为数学上有趣所以写一下。

Continuum hypothesis (CH):

$$2^{\aleph_0} = \aleph_1 \tag{37}$$

这是说：连续统 $[0, 1]$ 的基数 $2^{\aleph_0}$ 紧接在 可数集合的基数 之后。

1878 年，Cantor 提出 CH
1900 年，Hilbert 列出 连续统假设 为「23 问题」的第一个
　　Hilbert 给出了一个证明，但里面有 bug
1938 年，Gödel 证明 ZF + CH is consistent，换句话说：ZF cannot disprove CH
1963 年，Paul Cohen 证明 ZF cannot prove CH
　　他用的方法叫 "forcing"

Paul Cohen (1934-2007)

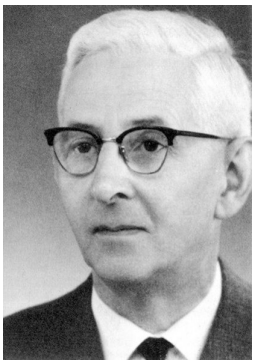
4.10 Kleene realizability

5 Intuitionistic logic

In 1933, Gödel proposed an interpretation of intuitionistic logic using possible-world semantics.

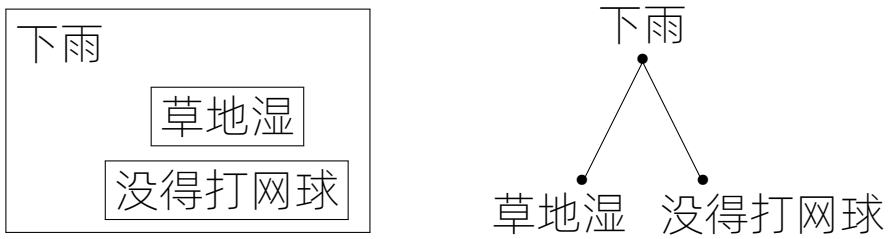
In topos theory $A \Rightarrow B$ is adjoint (via the hom-product adjunction) to $A \vdash B$, which is "okay" because it is independent of which implication (material or strict) we are using.

5.1 Heyting algebra

Arend Heyting (1898-1980)  (38)

1930 年，Heyting 给出了 constructive mathematics 的一种 axiomatization, called **intuitionistic logic** (IL). Heyting algebra 是一种 IL 的 **代数模型**，正如 Boolean algebra 是 经典逻辑的 代数模型。

例如，以下的 **拓樸**模型 和 **格** (lattice) 模型，都是 Heyting algebra 的模型：



(39)

两者之间是等价的，起源於 **Stone duality** (every Boolean algebra is isomorphic to a topology of open sets)，其后再被推广到 **Priestley** 拓扑对偶 等，都是大同小异的。

注意：下雨 $\rightarrowtriangle$ 草地湿 是 Heyting implication，这个 $\rightarrowtriangle$ 的存在 并不是因为「下雨」的**真值** 比「草地湿」的**真值 小**。

The Heyting implication $a \rightarrowtriangle b$ exists for all elements a, b, x such that:

$$x \leq (a \rightarrowtriangle b) \quad \text{iff} \quad (x \wedge a) \leq b. \tag{40}$$

Every Boolean algebra can be a Heyting algebra with the material implication defined as usual: $a \Rightarrow b \equiv \neg a \vee b$.

Heyting algebra is to intuitionistic logic what Boolean algebra is to classical logic. But this may not jibe with the idea of "strict implication".

In a topos $\mathbb{E}$, the subobject $\mathrm{Sub}_{\mathbb{E}}(A)$ is a **poset** that admits **Heyting implication**.

Using Kripke semantics, the Heyting arrow $\to$ can be defined by:

$$k \Vdash A \to B \quad \Leftrightarrow \quad \forall \ell \geq k \; (\ell \Vdash A \Rightarrow \ell \Vdash B) \tag{41}$$

Whereas the "fish-hook" **strict implication** can be defined as "A implies B necessarily":

$$A \multimap B \quad \equiv \quad \Box(A \Rightarrow B) \tag{42}$$

The two can be regarded as equivalent via:

$$\begin{aligned} k \Vdash \Box(A \Rightarrow B) \quad &\Leftrightarrow \quad \forall \ell \geq k \; (\ell \Vdash (A \Rightarrow B)) \\ &\Leftrightarrow \quad \forall \ell \geq k \; (\ell \Vdash A \Rightarrow \ell \Vdash B) \end{aligned} \tag{43}$$

问题是将 Heyting implication 定义 by possible worlds semantics 有什么好处? 从 machine learning 角度可能更合理。但其实也好像包含 classical implication 所以是循环的?

根据 topos 理论, Heyting implication 的出现是因为 sub-objects 的 Heyting algebra. 但我希望 implication arrow 纯粹是因为 BHK interpretation 而出现的。

6 Modal logic

Modalities are often conceived in terms of variation over some collection or **possible worlds**.

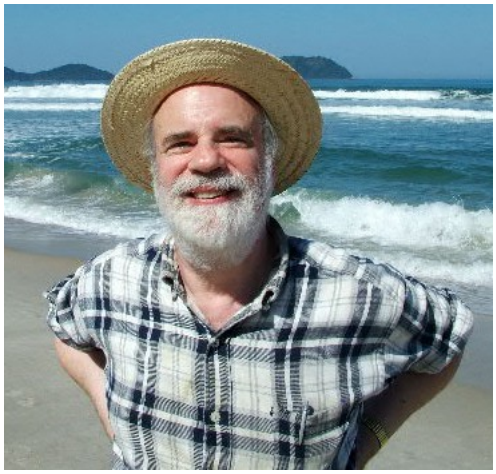
A modal operator (such as $\Box$) in the category $\mathbf{Sheaf}(X)$ is a **sheaf morphism** $\Box : \Omega \to \Omega$ satisfying 3 conditions, $\forall U \subseteq X$ and $p, q \in \Omega(U)$:

$$\begin{aligned} &\text{a)} \quad p \leq \Box(p) \\ &\text{b)} \quad (\Box;\Box)(p) \leq \Box(p) \\ &\text{c)} \quad \Box(p \wedge q) = \Box(p) \wedge \Box(q) \end{aligned} \tag{44}$$

6.1 Possible-world semantics

Possible-world semantics is also called **intensional semantics**, as opposed to **extensional semantics** where truth values are directly assigned to propositions. Does this idea jibe with the other definition of "intension", ie, as opposed to Leibniz extensionality and also related to intensional logic?

Saul Kripke (1940-)



6.2 Computer implementation of possible worlds

要詮釋 modal logic，需要引入 frame $F = \langle W, R, D, H \rangle$, 其中：

- W = set of possible worlds = $\{w_1, w_2, ...\}$
- R = a relation between worlds, $w_i R w_j$
- D = domain of first-order objects
- $H : W \to \wp(D)$, for each world specify a subset of objects

To interpret formulas with $\Box$:

$$M \Vdash \Box A\,[w] \quad \Leftrightarrow \quad \forall w' \succeq w.\ M \Vdash A\,[w'] \tag{45}$$

這牽涉到要 quantify over all w's.

重點是 inference 需要什麼 data?

顯然需要決定在某 w 中 p 是否成立，甚至需要判斷 $\forall w.p[w]$.

後者似乎需要將所有 有關的可能世界（或至少是其 summary）放到 working memory 再 quantify.

6.3 Intensional vs extensional

"Beethoven's 9th symphony" and "Beethoven's choral symphony" has the same **extension** but different **intensions**.

6.4 Intensional logic

Possible-world semantics is also called **intensional semantics**, as opposed to **extensional semantics** where truth values are directly assigned to propositions.

Logic terms differ in intension if and only if it is **possible** for them to differ in extension. Thus, **intensional logic** interpret its terms using possible-world semantics.

6.5 Strict implication

The problem of "material implication"

Material implication 的意思是「实质蕴涵」，亦即是说 $A \Rightarrow B$ 等价於 $\neg A \vee B$，其真值表如下：

A	B	$A \Rightarrow B$
0	0	1
0	1	1
1	0	0
1	1	1

(46)

Material implication 的概念向来很有争议，例如，当前提是错误时，它永远是真的：

$$\text{瑞士在非洲} \Rightarrow \text{猪会飞} \tag{47}$$

透过观察 truth table 可以发现，它的每一列 其实代表一个 **可能世界**，这些 可能世界 是不会 同时发生的。换句话说，material implication 和 strict implication 本来是一样的，只是前者将 可能世界 的语义 隐蔽到「幕后」。

For strict implication to make sense, it is always necessary to invoke possible-world semantics. A strict implication is always **learned** from numerous examples from experience, in accord with the philosophical tradition of "empiricism".

Strict implication is equivalent to material implication over multiple instances. The truth table of material implication agrees with the functional interpretation of implication.

7 Fuzzy logic

Lotfi Zadeh (1921-2017)

Iranian-Jewish

首先是 implication 的问题，fuzzy implication 并不对应於 material implication in Boolean algebra.

另外有个问题就是要考察一下 fuzzy truth value 在各种情况下的正确性。

例如假设 set 里面有 fuzzy proposition 的「证明」
又或者「人类」的集合是「有人类」这个命题的证明
而，「数学家」作为「人类」的子集，等於命题「所有人都是数学家」的证明
而这和 fuzzy value 是一致的

但为什么「John 是人」这个命题有点怪怪的？
如果它有 fuzzy value，应该是某集合的子集
有些元素证明 John 是人，有些证明他不是人
或者是 John 的**属性**的集合？

而其中有些属性 imply 他是人？
或者有些属性 $\subseteq$ 人的属性？

还有这跟 “Marilyn Monroe is sexy” 是不是一致？Marilyn 的所有属性集合，其中 imply sexy 的 subset
还是 sexy 的所有属性集合，其中 Marilyn 也有的？

Sexy(marilyn), Human(john), vs Human(Mathematicians).

What kind of mapping does this require?

7.1 Fuzzy implication

Implication 能不能 generalize 到 fuzzy logic 的情况？

7.2 Fuzzy functions?

What are fuzzy functions?

8 Homotopy type theory (HoTT)

Vladimir Voevodsky (1966-2017)

HoTT 的中心思想是将 types 看成是某些 空间 (spaces)，而这些空间可以被赋予 topological 特别是 homotopy 结构。在 homotopy 而言，关键是 将 type A 里面两个元素的 **相等** id_A 看成是 homotopy 的 **path**.

8.1 HoT levels

...	...
2	2-groupoids
1	groupoids
0	sets
-1	(mere) propositions
-2	contractable spaces

(48)

"Truncation"

8.2 What is homotopy?

8.3 Univalence axiom

在 HoTT 的 set 的层次，"=" 是一个 predicate, 根据我的理论可以看成是 fuzzy predicate.

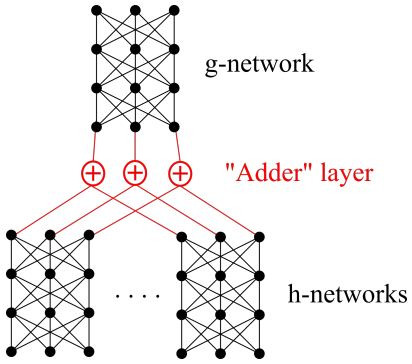
如果我没理解错误，univalence axiom 的意思 似乎是将 = 的 fuzzy truth value 「强制」成 binary.

9 Transfer to deep learning

这一节讨论 如何将上面提到的 logic structure 转移到 深度学习的 神经网络 (neural network, NN)。一般来说，很难将 额外的结构 impose 在 神经网络上，因为 NN 本身已经有很 "rigid" 的结构。在这方向上成功的例子是 CNN (convolutional NN)，其中 convolution $f * g$ 是由 "weight sharing" 模拟。但除此之外，一般很难在 NN 上添加结构。

Symmetric NN 是一种 在输入变量的 **置换**下 不变 (permutation invariant) 的神经网络。其解决 办法 是利用以下的 函数形式：

$$f(x, y, ...) = g(h(x) + h(y) + ...) \tag{49}$$



(50)

注意在这个方案下，NN 是以 “black box” 作为 building blocks；内部结构不变。

范畴论 的好处是，将一切用 morphisms, compositions, pullbacks, adjunctions, 等 表示；这种做法 很容易用 NN implement.

9.1 Propositional aspect

在命题逻辑的层次上，我选择了最简单的特性，亦即是命题之间的 commutativity:

$$\begin{array}{ccc} A \wedge B & \equiv & B \wedge A \\ \text{下雨} \wedge \text{失恋} & \equiv & \text{失恋} \wedge \text{下雨} \end{array} \tag{51}$$

但我没有 fully exploit Heyting algebra or Boolean algebra 的结构，因为可以预见 将来是会 推广到 fuzzy-probabilistic logic，而后者的结构只需要 $\wedge$ 和 $\Rightarrow$，和 binary logic 有些不同（迟些解释....）

在 命题层面 还有一个很重要的 **product-hom adjunction**, 它说的是 $A \wedge B$ 和 $A \Rightarrow B$ 之间的邻接：

$$(A \times B) \to C \quad \simeq \quad A \to (B \to C) \tag{52}$$

这在逻辑上 是见惯的，没有什么稀奇，但它推论 $A \Rightarrow B$ 可以替代 $A \vdash B$，由於 $\vdash$ 在我们的 AI 系统中是 **神经网络**，这表示 神经网络中的「黑箱」知识可以「外在化」(externalize) 成 **逻辑命题**，这一点 在智能系统中 是有关键的重要性，因为它表示 知识可以透过语言学习得到（虽然这也不是必需 范畴论 才可以看得出来。）

9.2 Predicate aspect

目前，一般的深度学习模型是比较 简单 / 粗暴 地作用在 自然语言句子的 **syntax** 层面，例如：

$$\text{“Je } \bullet \text{ suis } \bullet \text{ étudiant”} \xrightarrow{f} \text{“I } \bullet \text{ am } \bullet \text{ student”} \tag{53}$$

句子中的 words 是用 **Word2Vec** 方式 embed 到向量空间。但如果用了 Curry-Howard 对应，则会有些微不同，而这个微妙的差异 在数学上比较完美，实际上 computer implementation 会不会比较优胜？再看一次 Curry-Howard correspondence:

$$\begin{array}{c} A \Longrightarrow B \\ \hline \blacksquare \xmapsto{f} \blacksquare \end{array} \tag{7}$$

这里 A 和 B 是**逻辑命题**，例如「我是学生」；f 将 A 命题里面的 witness $\blacksquare$ 映射到 B 命题里面。注意：「我是学生」这些**语法**上的信息，是以 f 的 **domain** 和 **co-domain** 表示的。

这一节 我们要讨论的是 命题及其内部 在计算机上的 implementation 问题。再看一次 图 (25) 的 两个层次：

$$\overbrace{\underbrace{\text{Human (Socrates)}}_{\Omega}}^{A} \Rightarrow \overbrace{\underbrace{\text{Mortal (Socrates)}}_{\Omega}}^{B} \tag{25}$$

红色 $\rightarrow$ 是 **predicates** 形成的层次。A 和 B 是两个不同的**空间**。在 A **内部**，Socrates 也是一个空间，Human 是另一个空间，而 Human(Socrates) 构成新的空间（透过 dependent type constructor Σ）。

假设 $X \in \mathcal{U}_1, P \in \mathcal{U}_2, P(X) \in \mathcal{U}_3$，（这些 $\mathcal{U}_i$ 是 type universes），在计算机上最简单的做法是令：

$$\mathcal{U}_3 = \mathcal{U}_1 \times \mathcal{U}_2 \tag{54}$$

这和 §3.5 说的 Σ type constructor 本质上是 Cartesian product 的说法吻合。

到此，我们发现，用 Curry-Howard 的做法 和「粗暴」的 syntactic 做法其实是一模一样的！有点失望，但我希望这些 漂亮的数学 不会是完全无用的....

9.3 Modal aspect

References

欢迎提问和讨论 ☺